Hindawi Publishing Corporation
International Journal of Computer Games Technology
Volume 2010, Article ID 979235, 8 pages
doi:10.1155/2010/979235

Research Article

Motion in Augmented Reality Games: An Engine for Creating Plausible Physical Interactions in Augmented Reality Games

Brian Mac Namee, David Beaney, and Qingqing Dong

DIT AI Group, Dublin Institute of Technology, Dublin, Ireland

Correspondence should be addressed to Brian Mac Namee, brian.macnamee@dit.ie

Received 15 December 2009; Accepted 12 July 2010

Academic Editor: Soraia Musse

The next generation of Augmented Reality (AR) games will require real and virtual objects to coexist *in motion* in immersive game environments. This will require the illusion that real and virtual objects interact physically together in a plausible way. The *Motion in Augmented Reality Games* (MARG) engine described in this paper has been developed to allow these kinds of game environments. The paper describes the design and implementation of the MARG engine and presents two proof-of-concept AR games that have been developed using it. Evaluations of these games have been performed and are presented to show that the MARG engine takes an important step in developing the next generation of motion-rich AR games.

1. Introduction

Augmented Reality (AR) is now a mainstream technology, as evidenced, for example, by the recent release by the Topps Company, Inc (http://www.toppstown.com/) of AR baseball cards featuring animated 3D players. Building on its use in advertising, AR technology is expected to be used extensively in games, and there have already been a number of efforts in this regard. Even more so than conventional games, AR games require sophisticated physics simulation, as the purpose of any AR application is to create the illusion that real and virtual objects coexist. This requires the addition of realistic physical motion to allow real and virtual objects to interact together in a way that appears plausible.

The contributions of this paper are a description of the design and implementation *Motion in Augmented Reality Games* (MARG) engine which has been developed to bring physically realistic and plausible motion to AR games, and descriptions of evaluations of two proof-of-concept games that have been developed using the engine. Before discussing these contributions Section 2 will describe related work, particularly focusing on AR applications (some of which are games) that have used physics simulation. Section 3 will then describe the MARG engine, and this will be followed in Section 4 by a description of two games that have been developed using the engine, and their evaluation. Finally, Section 5 will summarize how the use of the MARG engine improves AR games and outline directions for future work.

2. Physical Motion in Augmented Reality Games

AR applications augment a user's view of a real environment with virtual objects in order to create novel and immersive experiences. Successful applications have been developed in the military, industrial, medical, and advertising domains amongst others. There have also been some notable AR games which include ARQuake [1], an AR game that extends the famous computer game Quake4; Phone Tennis [2], in which player's use their mobile phones as tennis racquets; the Eye of Judgment (http://www.eyeofjudgment.com/), a card fighting game in the vein of Pokemon in which 3D animated monsters appear on top of the cards when viewed through a Sony PlaystationEye camera, the first commercially available AR game.

An important emerging aspect of AR research is creating seemingly physical interactions between real and virtual objects. Managing these interactions is a significant challenge as the impact of real objects on virtual ones must be simulated, and unrealistic or unbelievable physical interactions between objects can unrecoverably break the illusion created, which is unacceptable [3]. Adding these kinds of interactions to AR systems offers the potential to greatly improve the sense of immersion created for users, and will be necessary for creating the next generation of motion-rich AR games.

There are a number of examples of AR applications that combine real and virtual objects—albeit to different levels of sophistication. Three early systems are the games *STARS* [4], *Monkey Bridge* [5], and the *Invisible Train* [6] (which is possibly more of a toy than a game). STARS is a hardware and software system which allows the creation of AR board games in which real and virtual pieces exist side-by-side on the game board. Monkey Bridge is a game in which the player must build a physical bridge from a set of prototype bridge pieces. When viewed through a head-mounted display (HMD) a virtual character appears to cross the bridge if it is built correctly. Similarly, in the Invisible Train system, a virtual train appears to drive around real tracks when viewed through a suitable mobile device such as a smart phone. However, in all of these systems, although real and virtual objects appear to exist together, there is no simulation of the physical interactions between them.

There have, though, been a number of AR applications which use realistic physics simulation. Two of the most impressive were developed as physics teaching aides. The PhysicsPlayground [7] simulates physics experiments viewed through an HMD. The experiments appear to take place on a clipboard-type pad and can be manipulated using a light pen. The Interesting Mechanism [8] takes a more playful approach and is a table-top AR application in which users build virtual Rube Goldberg machines (Rube Goldberg machines are machines designed to do simple tasks in hugely convoluted ways and were made famous by the cartoons of American cartoonist, sculptor, author, engineer, and inventor Reuben Lucius Goldberg [9].) to achieve a range of challenges. Again, users view this system through an HMD and the machines built from predefined virtual pieces are physically simulated.

Another notable example of the use of physics and AR is the Goblin XNA [10] system that has been developed as a framework for creating AR applications using the Microsoft XNA framework (http://www.msdn.microsoft.com/en-us/xna). As a demonstration of this system an AR racing game has been built in which users drive a virtual racing car around a table top environment viewed through an HMD. The behaviour of the car is physically realistic and it interacts believably with virtual objects. While physics simulation is used in all of these examples, it is only used for interactions between virtual objects. AR applications in which real and virtual objects interact believably together are more rare, but there are some examples.

One of the most impressive AR applications in which real and virtual objects appear to interact together is the IncreTable system [11]. In this system users play games with real and virtual objects on top of a table that encloses a projector through which virtual objects are displayed from underneath. The real objects that can be used in the system include everyday objects such as books (which are viewed as physical obstacles), dominoes, special *interaction objects* that move interactions between the real and virtual world, and robots which can be driven by the user and can be used to knock over both real and virtual dominoes. However, the interactions between real and virtual objects, although impressive, are relatively limited. Physical interactions between real and virtual objects are restricted to two dimensions and often the only way real and virtual objects can appear to interact together is through special portal objects.

Two other systems are worth considering. The work of Lok et al. [12] allows a range of interactions between real and physical objects—for example, a user can use their hands to seemingly open a set of virtual curtains or real blocks can be used to effect the path of a virtual ball. In the *Kobito: Virtual Brownies* [13] system virtual objects appear to physically affect real objects—breaking an apparently impossible boundary. In this system small virtual characters appear to push along a small robotic tea caddy.

While all of the examples described above are impressive, there is still a long way to go for AR applications to sufficiently achieve a physically plausible blending of real and virtual objects. Interactions must take place freely in three dimensions and not be limited to a small number of special objects. Rather all objects used within a game world—both real and virtual—should appear to interact physically. Furthermore, the interactions between real and virtual objects must be physically plausible if the illusion that these objects coexist is to be successfully created, and so the level of physical simulation used must be suitably sophisticated. These are the issues that the *Motion in AR Games* (MARG) engine has been designed to address.

3. The Motion in Augmented Reality Games Engine

In order to create games in which real and virtual objects appear to interact physically together the *Motion in Augmented Reality Games* (MARG) engine has been developed. The main components of this are shown in Figure 1 and are a *World Model*, an *AR Registration Engine*, a *Physics Engine*, and a *Renderer*. The arrows in the diagram show how information flows between these components. This section will describe each of these components in detail. The discussion assumes a generic hardware setup in which a camera views a real environment, images of which are shown to a user on some display device with augmented virtual objects. Specific details of the hardware setups used in the games developed will be discussed in Section 4.

3.1. The World Model. Within the MARG engine the World Model maintains a list of all of the objects within a game environment. These objects can be real, virtual, or composites of real and virtual objects (e.g., a real tank vehicle with a virtual turret). Figure 2 shows a class diagram of the different kinds of objects that can be represented within the MARG engine. The key features of this diagram are

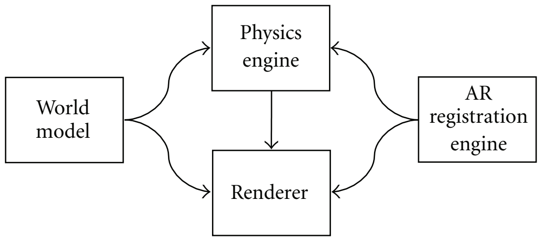

Figure 1: The components of the Motion in Augmented Reality Games engine.

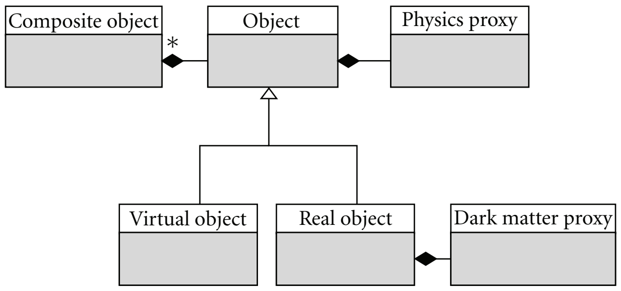

Figure 2: A class diagram of objects used in the MARG engine.

(i) that a composite object can be made up of a set of real and virtual objects,

(ii) that every object (both real and virtual) has a *physics proxy* associated with it for use in the Physics Engine (discussed further in Section 3.3),

(iii) that each real object also has a *dark matter proxy* associated with it for use by the Renderer (discussed further in Section 3.4).

3.2. The AR Registration Engine. All AR systems need a means through which to *register* virtual objects with the real environment. Registration involves the determination of the coordinate transformation required to make virtual objects appear as if they are in the desired location with respect to the real environment, based on the position of the camera through which the real environment is being viewed. Even small errors in registration can have a devastating effect on the illusion of immersion created in AR systems.

This MARG engine uses *fiducial marker tracking* [14] to perform registration (although any other registration technique could be used), and in the proof-of-concept games the ARToolkit API (http://www.hitl.washington.edu/artoolkit/) is used. Fiducial markers are easily recognized patterns which are placed within a real environment in order to assist with registration. Sample fiducial markers can be seen in Figure 4. The thick black border allows markers to be easily recognized within an image, and for the transformations between the markers and the camera to be calculated (based on the appearance of what is known to be a square border). The pattern within the centre of each marker allows it to be uniquely identified. Markers are used for two purposes in MARG: to register virtual objects within the real world, and to locate known physical objects.

The AR registration component in the MARG engine searches each camera frame for fiducial markers, the transformations to which are passed to the Renderer and Physics Engine components of the engine.

3.3. The Physics Engine. In order to create AR games in which real and virtual objects appear to inhabit the same physical space and perform plausible physical interactions, a physics engine is required. This is used to simulate the motion and interactions of virtual objects within a scene, and to simulate the interactions between real and virtual objects. So, for example, if a real robotic forklift appears to drive into a pile of virtual crates, the crates should fall over realistically, bouncing of the forklift and each other as appropriate.

Simulating the interactions between virtual objects requires only the straight-forward use of a physics engine, in which each virtual object is represented (using a simplified model) and simulated in the engine. (The detailed workings of modern physics engines are beyond the scope of this article. A good overview can be found in [15].) However, simulating the interactions between real and virtual objects is a little more complex. Within MARG each real object is given a *physics proxy* which is used within the physics engine to simulate the interactions between that object and the virtual objects within the scene. Figure 3(b) shows an example of a real robotic forklift used within one of the games developed using MARG (discussed in Section 4.2) and its physics proxy. The forklift is represented through a set of rigid bodies that approximate its shape and size. A physics proxy object must be handcrafted for each real object (or type of object) that will be used within a game. Although never rendered, the proxy objects are used in the Physics Engine to simulate interactions between the real object and virtual objects within the same scene.

In the MARG engine the positions of real objects are determined by locating fiducial markers placed on them. This requires a translation between the coordinate reference frames used by the Physics Engine, the AR Registration Engine, and the Renderer which is achieved by using the camera position as a common reference point between the systems.

Synchronizing the position and orientation of the physics proxies used to represent real objects within the Physics Engine directly with the movements of their associated fiducial markers is an attractive solution to simulating their movement within the Physics Engine. This approach, however, does not work well in practice and in fact can lead to chaos. When the fiducial marker associated with a real object is momentarily lost the physics proxy representing the object can be instantly transported across relatively large distances—often to positions *inside* other entities—when the marker is found again.

A more appropriate solution is the use of *key-framing* within the Physics Engine, which is similar to key-framing in animation. A key-framed physics entity within the Physics Engine is entirely under the control of the programmer, and has an effectively infinite mass. These entities go only where they are instructed to go, and are not affected by collisions

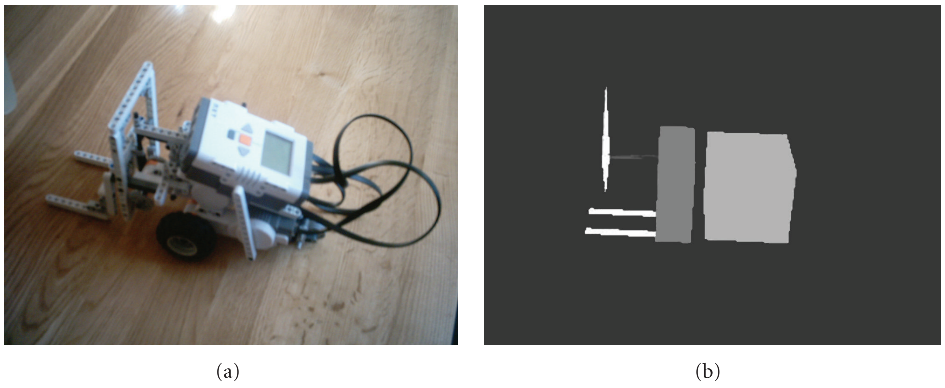

Figure 3: A real robotic forklift and its physics proxy.

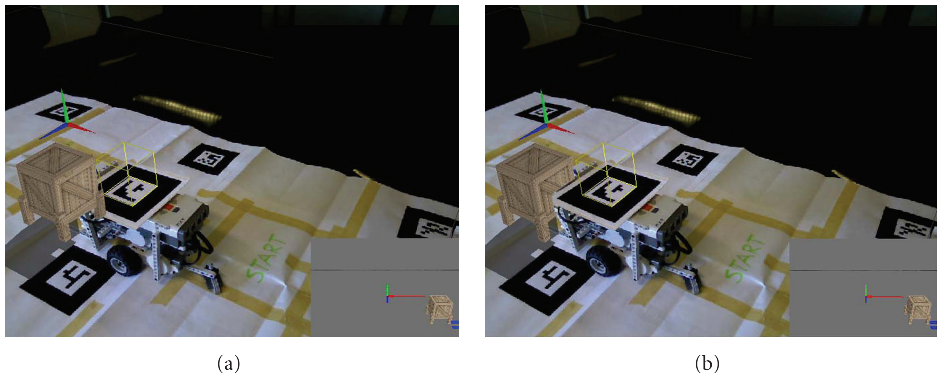

Figure 4: The same view of the real forklift and virtual crates with no occlusion and with dark matter based occlusion turned on.

with other entities or gravity—which is appropriate as it is not yet possible to have real objects reflecting these. Once a new position for a real object is given to the Physics Engine by the AR Registration Engine (based on the recognition of the appropriate fiducial marker), the Physics Engine calculates the path that the key-framed physics proxy for this real object must take to get to the new position, moves the proxy along this path (over a specified amount of time), and simulates interactions with any virtual objects encountered along this path. The actual implementation of the PhysicsEngine in the MARG engine is technology agnostic and the proof-of-concept games described in Section 4 use both the Open Dynamics Engine (http://www.ode.org/) and Havok (http://www.havok.com/) physics engines.

3.4. The Renderer. In rendering AR scenes believable *occlusion* (i.e., ensuring that when virtual objects move behind real ones they are suitably cropped) is extremely important. Figure 4 shows an example from one of the games described in Section 4.2 in which a real robotic forklift is shown interacting with virtual crates with and without occlusion.

In the MARG engine an approach based on *dark matter* [16] is used. Invisible proxy rendering objects are drawn to the *depth* (or *z*) *buffer* used by the renderer to represent real objects before virtual objects are drawn. The result of this is that although the proxy objects cannot be seen, when virtual objects behind them are drawn by the renderer they are suitably cropped as the renderer believes there is another object drawn in front already. While the dark matter approach creates an adequate illusion of occlusion it does have limitations in that the proxy objects must be designed to closely fit the real objects (with positioning based on the recognition of AR tags), and that the illusion can be broken through the introduction of an unexpected object such as a user's hand.

Other than occlusion there are many rendering techniques that can add to the illusion created in AR games such as the use of shadows and real-time lighting effects, but these are outside the scope of this paper. Again, the Renderer component in MARG is technology agnostic, however, all of the proof-of-concept games use OpenGL (http://www.opengl.org/) for rendering.

4. Proof-of-Concept Games

To validate the use of the MARG engine two AR games in which physical interactions between real and virtual objects are a key feature have been developed. These are both table-top games, the first—*Table-Top AR Racing* [17]—is a car game in which a virtual car drives around the player's desk interacting with real obstacles, while in the second—*Forked!* [18]—the player must control a real robotic forklift to manipulate virtual crates. This section will describe these games in detail.

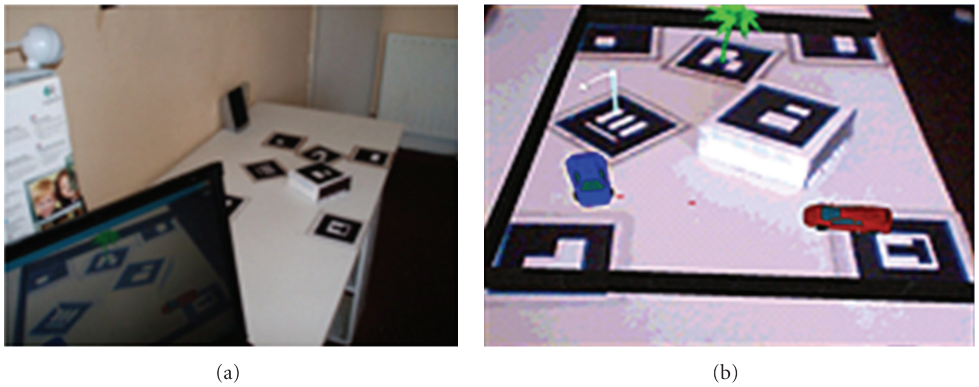

Figure 5: The setup of the table-top car racing game and two screenshots of the game viewed with augmentations.

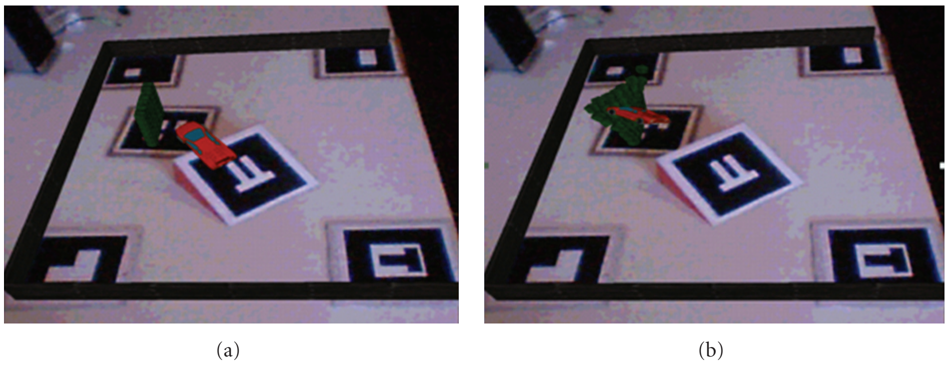

Figure 6: A virtual car interacting with a real ramp in the Table-Top AR Racing game.

4.1. Table-Top AR Racing. In the first proof-of-concept game, virtual cars appear to drive around the player's desk, interacting with each other and a range of virtual and physical objects. Figure 5 shows the hardware setup used (a camera looks down at the desk and the scene, with virtual augmentations, is displayed on a monitor positioned at the desk's edge) and a screenshot of what players see. Virtual objects in the game include walls, trees, lampposts, piles of boxes, and see-saws; each of which is represented by a particular fiducial marker and simulated in the physics engine. Most of these virtual objects are tied to the fiducial markers representing them (e.g., the trees and lampposts), but others are free to move away from the fiducial markers after a game begins (e.g., the cars themselves or the boxes that start in a pile, but move away from this when hit by the car as shown in Figure 6).

Where the game becomes particularly interesting is when real objects are introduced and allowed to interact with the virtual ones. The large box in the middle of the desk in Figure 5(b) is an example. The box has a fiducial marker placed on top of it which is recognized by the AR Registration Engine. A physics proxy for this box, based on preprogrammed parameters that describe the shape and size of the box, is then used within the Physics Engine to calculate interactions between this and virtual objects. So, for example, the virtual cars will crash if driven into the box. The game also features a physical ramp (modeled in the same way as the box) that can be seen in Figure 6.

The game can be played in a number of different modes. The first is a free-flow *sandbox* mode in which players can experiment with different objects in order to explore how they interact. In the second mode, two players control virtual cars and are able to shoot each other. Real objects such as the block and the ramp are particularly important in this game mode as they can be used as cover from bullets. In informal trials of the game players reported that, although the game modes could be more compelling, they enjoyed the experience and in particular the interactions between real and virtual objects.

While this game successfully demonstrated how the MARG engine could be used to allow the creation of AR games in which real and virtual objects interact together, the interactions were not especially complex. The real objects used in the game were stationary and so their interactions with virtual objects were fairly simple. For this reason, a second game was developed in which much more complex interactions were required between real and virtual objects.

4.2. Forked! In *Forked!* the player drives a real robotic forklift (built using a Lego Mindstorms NXT (http://mindstorms.lego.com/) robotics kit) and is tasked with moving virtual crates around the environment. Again, the game can be played in a number of different modes, including a sandbox mode and a selection of mini-games in which the player must complete particular tasks (e.g. moving a crate between

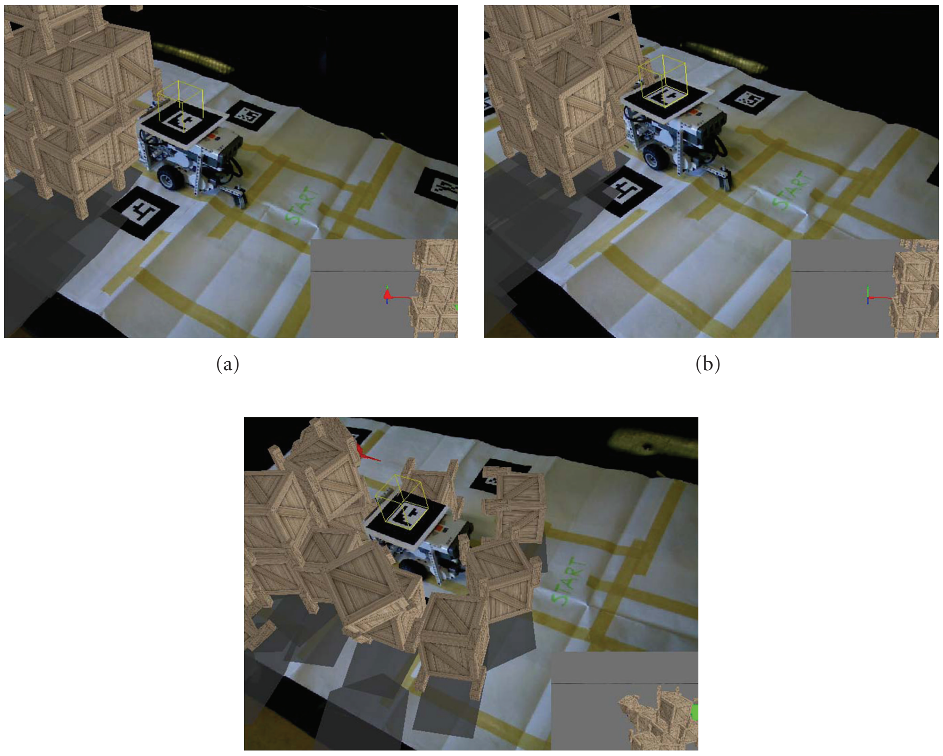
(a) (b) (c)

Figure 7: A series of snapshots of the real robotic forklift appearing to physically interact with a wall of virtual crates.

a set of goal posts). In the hardware setup for this game the player views the game world through a Trivisio ARVision 3D head-mounted display (http://www.trivisio.com/) and controls the forklift using an X-Box games controller (http://www.xbox.com/). For illustration, Figure 7 shows the forklift interacting with a stack of virtual crates (a comparison with Figure 6 is worthwhile).

The key difference between *Forked!* and Table-Top AR Racing is that the interactions between the real and virtual objects are much more subtle. Because the real forklift can move around and is tasked with interacting closely with the virtual crates, the sophistication of the interactions is more complex. For example, crates can be raised and lowered using the forks on the forklift, crates can be carried over distance on the forklift, the forklift can crash through multiple crates, and crates can be pushed around using the forklift. All of this is achieved successfully, as was established through a series of evaluation experiments performed using this game. These are described in detail in [18], but are summarized here.

The motivation for the evaluation experiments was to determine if the physical interactions between real and virtual objects present in the game were realistic and believable. There has been a growing interest in recent years in how best to evaluate AR systems [19–21] and it is widely accepted that determining the realism and believability of simulations is notoriously difficult due to the subjectivity of these questions [22]. For this reason the evaluations of *Forked!* were primarily based on the performance of participants in a series of tasks they were asked to complete using the robotic forklift. For each task, participants were asked to perform a *reality-based* version in which the forklift interacted with small cardboard crates, and an *augmented-reality-based* version in which the forklift interacted with virtual crates. The time taken for each participant to perform each task was recorded and the evaluation results are primarily based on this timing information.

The tasks that participants were asked to complete were: *pick-up-one*, which asked participants to drive the forklift to a suitable position and pick up a single crate; and *forklift football*, in which participants had to pick up a crate and move it between a set of goal posts. Reality-based and AR-based versions of each task were performed by each participant in a random order. The AR-based versions of the tasks were very close replicas of the reality-based tasks with crates of similar sizes and apparent weights and densities placed in the same positions.

Eleven participants took part in this evaluation experiment. These participants were aged between 20 and 25; 10 of the 11 were males; all were computer literate; none had direct experience of AR applications. For the analysis of the results, it should be noted that what is being sought is evidence that, in terms of realism and believability, the experiences of users in the AR-based tasks were *comparable* to their experiences in the reality-based tasks. Obviously it is not expected that the AR-based tasks be *more real* than the reality-based tasks!

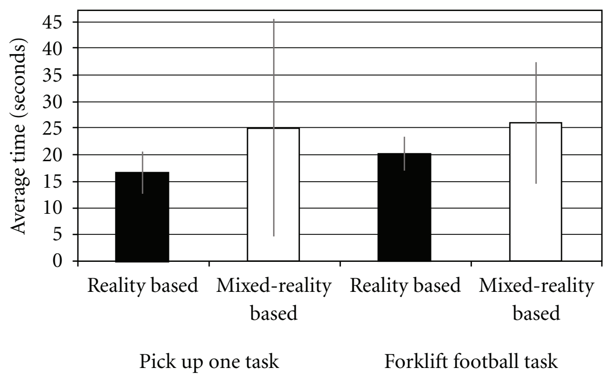

FIGURE 8: Average times taken (plus standard deviations) for subjects to complete the reality-based and AR-based versions of the pick-up-one and forklift football tasks.

On average it took participants 16.5 seconds to complete the reality-based pick-up-one task, 25.2 seconds to complete the AR-based pick-up-one task, 19.7 seconds to complete the reality-based forklift football task, and 25.9 seconds to complete the AR-based forklift football task. These average times and their standard deviations are shown in Figure 8.

From these figures it can be seen that the AR-based tasks took longer than their reality-based counterparts. It is also clear that there is much larger deviation in the AR-based tasks than in the reality-based tasks. However, while a paired two-tailed Student's t-test performed on the corresponding sets of time data for each task supports the fact that participants took longer on both of the AR-based versions of the tasks, this is not supported in either case with statistical significance. This is the key finding from this evaluation: *participants could complete the AR-based tasks in times comparable to those needed for the reality-based tasks*. That participants were able to perform tasks requiring sophisticated physical interactions between real and virtual objects in reasonable time provides strong evidence that these interactions were realistic and believable, and indicates that the MARG engine is appropriate for creating AR games featuring realistic motion including physically realistic interactions between real and virtual objects.

Section 5 will discuss the findings arising from these proof-of-concept games and outline the directions in which we intend to take this work in the future.

5. Discussion and Future Work

In order for the next generation of AR games to create the illusion that they are set in environments in which real and virtual objects coexist, these objects will need to exhibit plausible physical interactions. These interactions will need to take place in three dimensions, cannot be limited to a small number of special objects, and the level of sophistication of interactions between real and virtual objects must be high.

The first of these requirements is evident in both of the proof-of-concept games presented in Section 4. In *Table-Top AR Racing* the virtual cars can jump from real three dimensional ramps through highly stacked piles of virtual boxes. Similarly, in the screenshots of *Forked!* shown in Figure 7 the three-dimensional nature of the physical interactions is clearly evident.

The second requirement is that physical interactions between real and virtual objects not be limited to a small number of special objects. The MARG engine does not yet allow any real object to be included within the simulation of a game world, as physics and rendering proxies must be designed for each important object, and important objects must be recognised by the registration engine. Due to their simplified form, however, the design of physics and rendering proxies is not particularly time consuming (e.g., see the simplicity of the physics proxy shown in Figure 3(b)) and so large numbers of real objects can be easily incorporated into game worlds.

The third requirement is that the apparent physical interactions between real and virtual objects be simulated in a plausible and sophisticated way. This is best demonstrated in *Forked!* in which the real forklift interacts in subtle ways with the virtual crates. The evaluation presented in Section 4.2 further showed that participants were able to perform delicate operations involving interactions between the real forklift and virtual crates, and found these interactions to be comparable to those experienced when both the crates and the forklift were real. This level of plausibility and sophistication is beyond any of the examples described in Section 2.

In the near future, this work will be extended in the following ways. First, the AR Registration Engine will move away from the use of fiducial markers and instead use other generic pattern recognition techniques. Secondly, the Renderer will be elaborated upon to create more photo-realistic renderings of virtual objects. Finally, the Physics Engine will be extended first to allow more interesting interactions and more interestingly to allow virtual objects to physically impact real ones through the use of robotics as a bridge between the real and virtual worlds.

References

[1] W. Piekarski and B. Thomas, "ARQuake: the outdoor augmented reality gaming system," *Communications of the ACM*, vol. 45, no. 1, pp. 36–38, 2002.

[2] A. Henrysson, M. Billinghurst, and M. Ollila, "Face to face collaborative AR on mobile phones," in *Proceedings of the 4th IEEE and ACM International Symposium on Symposium on Mixed and Augmented Reality (ISMAR '05)*, pp. 80–89, Vienna, Austria, October 2005.

[3] J. Whitson, C. Eaket, B. Greenspan, M. Q. Tran, and N. King, "Neo-immersion: awareness and engagement in gameplay," in *Proceedings of the Conference on Future Play: Research, Play, Share (Future Play '08)*, pp. 220–223, Toronto, Canada, November 2008.

[4] C. Magerkurth, M. Memisoglu, T. Engelke, and N. Streitz, "Towards the next generation of tabletop gaming experiences," in *Proceedings of Graphics Interface*, pp. 73–80, London, Canada, May 2004.

[5] B. Istvan, M. Weilguny, T. Psik, and S. Dieter, "MonkeyBridge: autonomous agents in augmented reality games," in *Proceedings of the ACM SIGCHI International Conference on Advances in Computer Entertainment Technology (ACE '05)*, pp. 172–175, Valencia, Spain, 2005.

[6] D. Wagner, T. Pintaric, F. Ledermann, and D. Schmalstieg, "Towards massively multi-user augmented reality on handheld devices," in *Proceedings of the 3rd International Conference on Pervasive Computing (PERVASIVE '05)*, pp. 208–219, Munich, Germany, May 2005.

[7] H. Kaufmann and B. Meyer, "Simulating educational physical experiments in augmented reality," in *International Conference on Computer Graphics and Interactive Techniques (SIGGRAPH Asia'08)*, pp. 1–8, Singapore, December 2008.

[8] P. Buchanan, H. Seichter, M. Billinghurst, and R. Grasset, "Augmented reality and rigid body simulation for edutainment: the interesting mechanism—an AR puzzle to teach Newton physics," in *Proceedings of the International Conference on Advances in Computer Entertainment Technology (ACE '08)*, pp. 17–20, Yokohama, Japan, December 2008.

[9] P. C. Marzio, *Rube Goldberg: His Life and Work*, Harper & Row, 1973.

[10] O. Oda, L. J. Lister, S. White, and S. Feiner, "Developing an augmented reality racing game," in *Proceedings of the 2nd International Conference on Intelligent Technologies for Interactive Entertainment (ICST INTETAIN '08)*, pp. 1–8, Cancun, Mexico, 2008.

[11] J. Leitner, M. Haller, K. Yun, W. Woo, M. Sugimoto, and M. Inami, "IncreTable, a mixed reality tabletop game experience," in *Proceedings of the International Conference on Advances in Computer Entertainment Technology (ACE '08)*, pp. 9–16, Yokohama, Japan, December 2008.

[12] B. Lok, S. Naik, M. Whitton, and F. P. Brooks Jr., "Incorporating dynamic real objects into immersive virtual environments," in *Proceedings of the Symposium on Interactive 3D Graphics*, pp. 31–40, Monterey, Calif, USA, April 2003.

[13] T. Aoki, T. Matsushita, Y. Iio, et al., "Kobito: virtual brownies," in *Proceedings of the International Conference on Computer Graphics and Interactive Techniques (SIGGRAPH '05)*, pp. 1–11, Los Angeles, Calif, USA, 2005.

[14] M. Billinghurst, H. Kato, and I. Poupyrev, "The MagicBook: a transitional AR interface," *Computers and Graphics*, vol. 25, no. 5, pp. 745–753, 2001.

[15] I. Millington, *Game Physics Engine Development*, The Morgan Kaufmann Series in Interactive 3D Technology, Morgan Kaufmann, 2007.

[16] M. Fiala, "Dark matter method for correct augmented reality occlusion relationships," in *Proceedings of the IEEE International Workshop on Haptic Audio Visual Environments and Their Applications (HAVE '06)*, pp. 90–93, Ottawa, Canada, November 2006.

[17] Q. Dong, Z. Sun, and B. M. Namee, "Physics-based table-top mixed reality games," in *Proceedings of the 39th Conference of the International Simulation and Gaming Association (ISAGA '08)*, Kaunas, Lithuania, July 2008.

[18] D. Beaney and B. M. Namee, "Physical realism in augmented reality," in *Proceedings of the 8th International Symposium on Mixed and Augmented Reality (ISMAR '09)*, Orlando, Fla, USA, October 2009.

[19] A. Dünser, R. Grasset, and M. Billinghurst, "A survey of evaluation techniques used in augmented studies," in *International Conference on Computer Graphics and Interactive Techniques (SIGGRAPH Asia'08)*, pp. 1–27, Singapore, December 2008.

[20] M. Billinghurst, "Usability testing of augmented/mixed reality systems," in *Proceedings of the International Conference on Computer Graphics and Interactive Techniques (SIGGRAPH '08)*, pp. 1–13, Singapore, 2008.

[21] J. L. Gabbard and J. E. Swan II, "Usability engineering for augmented reality: employing user-based studies to inform design," *IEEE Transactions on Visualization and Computer Graphics*, vol. 14, no. 3, pp. 513–525, 2008.

[22] M. Slater, "How colorful was your day? Why questionnaires cannot assess presence in virtual environments," *Presence: Teleoperators and Virtual Environments*, vol. 13, no. 4, pp. 484–493, 2004.